Active learning of infectious disease epidemiology

Andreas Handel

University of Georgia

2019-07-10

The challenge

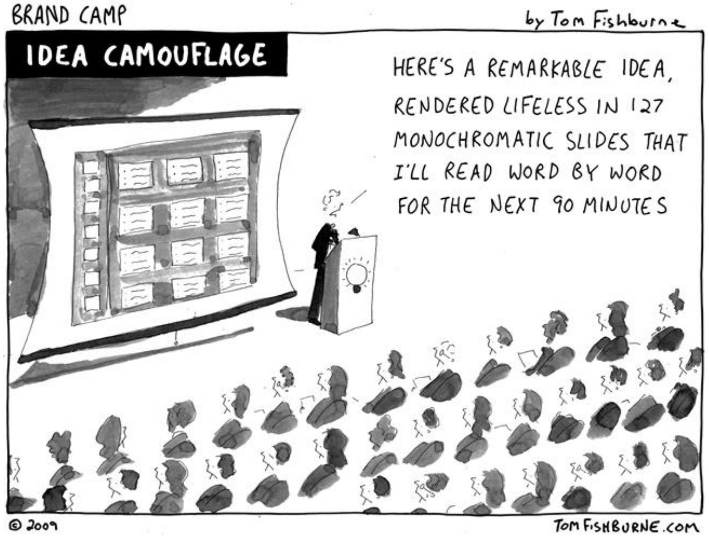
- Modern infectious disease epidemiology is fairly mathematical/computational.
- Models are generally implemented and analyzed on a computer.
- This requires students to use/write computer code.
- Many students have limited coding skills.
- The lack of coding skills can limit the use of models.

CAN'T YOU DO ANYTHING RIGHT?
GLASBERGEN
Copyright © Randy Glasbergen. www.glasbergen.com

www.glasbergen.com

The goals

- Help students learn modern, model-based, approaches to infectious disease epidemiology without having to write code.
- Allow for **active, hands-on learning.**
- Provide an (optional) way to easily progress toward increased coding.


BRAND CAMP
by Tom Fishburne
IDEA CAMOUFLAGE
HERE'S A REMARKABLE IDEA, RENDERED LIFELESS IN 127 MONOCHROMATIC SLIDES THAT I'LL READ WORD BY WORD FOR THE NEXT 90 MINUTES
© 2009
TOMFISHBURNE.COM

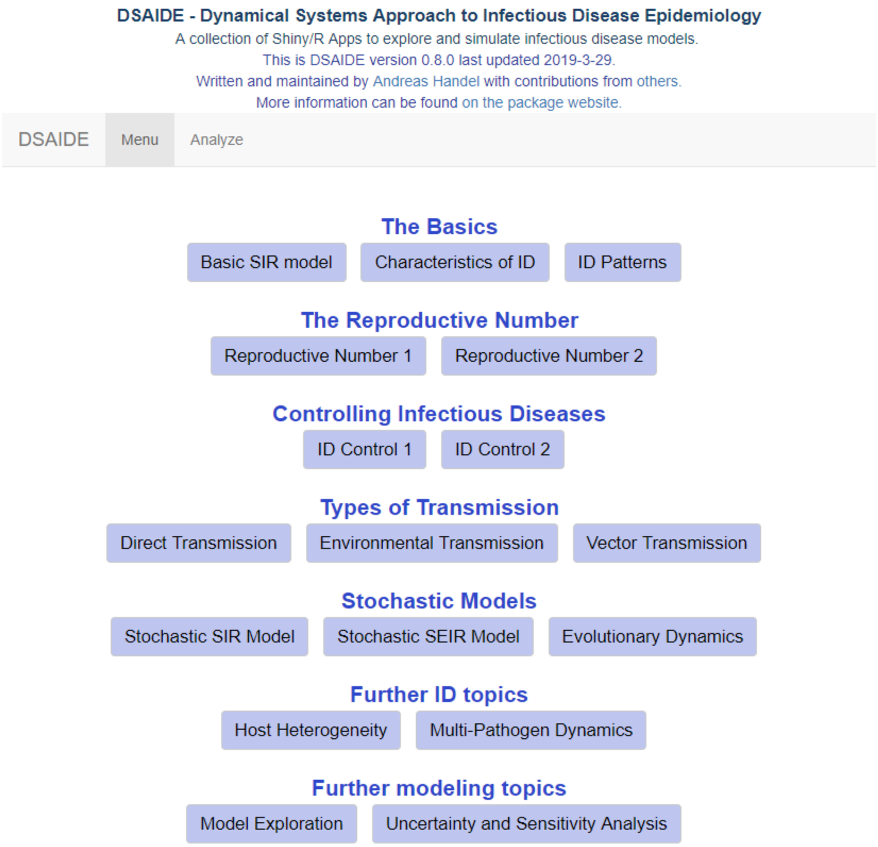
tomfishburne.com

A solution

- Write software that allows students to use models without having to write code.
    - DSAIDE - Dynamical Systems Approach to Infectious Disease Epidemiology: https://ahgroup.github.io/DSAIDE/
    - DSAIRM - Dynamical Systems Approach to Immune Response Modeling: https://ahgroup.github.io/DSAIRM/
    - See also: Handel 2017 PLoS Computational Biology *“Learning infectious disease epidemiology in a modern framework”*

DSAIDE overview

- Easy install like any other R package.
- Single command after package is installed and loaded to get to main menu.



DSAIDE - Dynamical Systems Approach to Infectious Disease Epidemiology

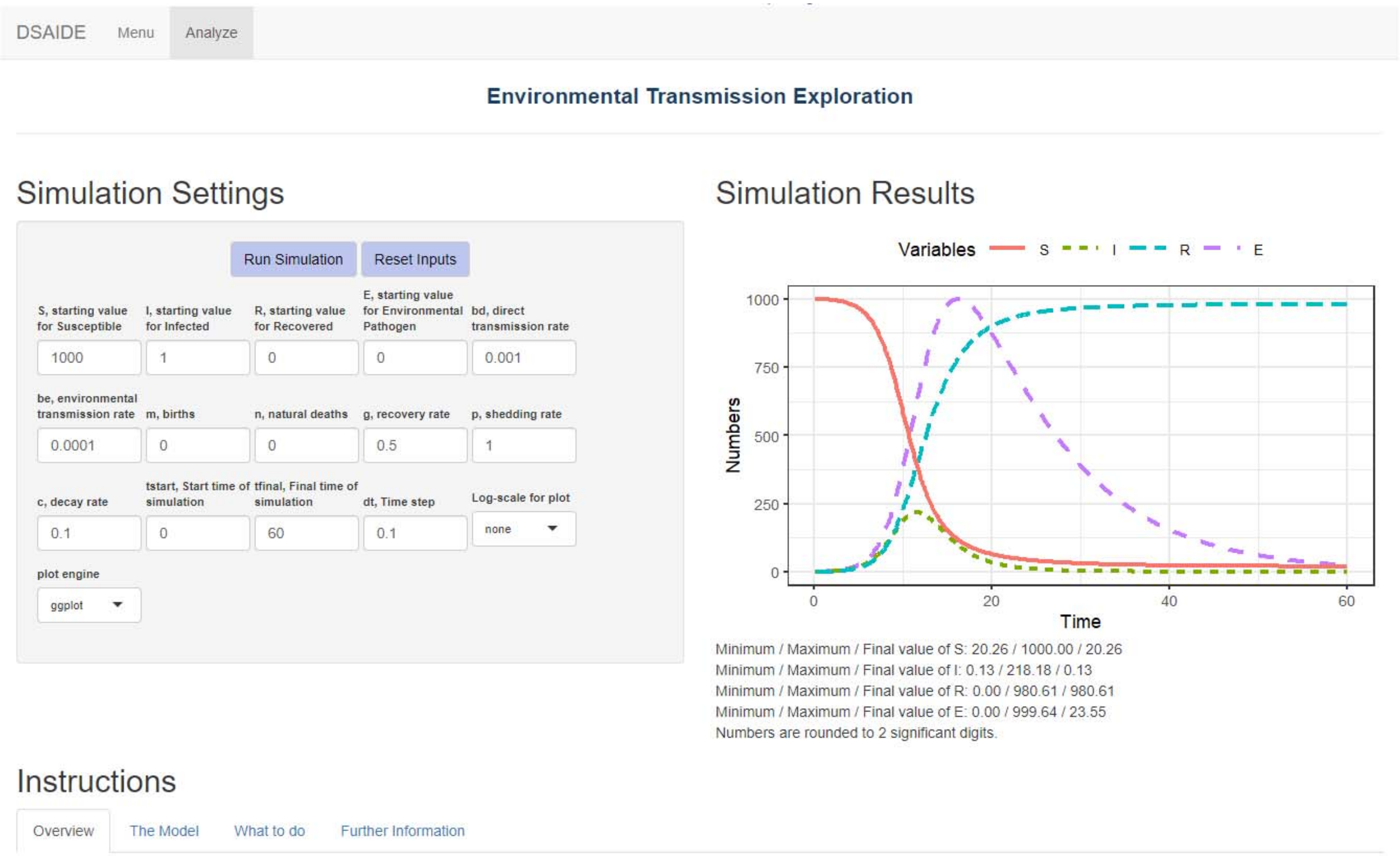
A collection of Shiny/R Apps to explore and simulate infectious disease models.

This is DSAIDE version 0.8.0 last updated 2019-3-29.

Written and maintained by Andreas Handel with contributions from others.

More information can be found on the package website.

DSAIDE interface



Environmental Transmission Exploration

Simulation Settings

S, starting value for Susceptible: 1000
I, starting value for Infected: 1
R, starting value for Recovered: 0
E, starting value for Environmental Pathogen: 0
bd, direct transmission rate: 0.001
be, environmental transmission rate: 0.0001
m, births: 0
n, natural deaths: 0
g, recovery rate: 0.5
p, shedding rate: 1
c, decay rate: 0.1
tstart, Start time of simulation: 0
tfinal, Final time of simulation: 60
dt, Time step: 0.1
Log-scale for plot: none
plot engine: ggplot

Simulation Results

Minimum / Maximum / Final value of S: 20.26 / 1000.00 / 20.26
Minimum / Maximum / Final value of I: 0.13 / 218.18 / 0.13
Minimum / Maximum / Final value of R: 0.00 / 980.61 / 980.61
Minimum / Maximum / Final value of E: 0.00 / 999.64 / 23.55
Numbers are rounded to 2 significant digits.

Instructions

DSAIDE documentation

In the *Introduction to ID* app, you explored a simple 3-compartment model, the basic SIR model. The model for this app has a few additional compartments, which allows us to include more details/realism into our model. We again focus on tracking individuals with regard to their infection/disease status. For this model, we track the following compartments/stages:

- **S** - susceptible, uninfected individuals
- **P** - presymptomatic individuals who are infected and do not yet show symptoms. Those individuals can potentially be infectious.
- **A** - asymptomatic, infected individuals. Those individuals can potentially be infectious.
- **I** - individuals who are infected and show symptoms. Those individuals are likely infectious, but the model allows to adjust this, including no infectiousness.
- **R** - recovered/removed individuals. Those individuals have recovered and are immune.
- **D** - individuals who have died due to the disease.

Of course, as with the basic SIR model, we could include further details by extending the number of compartments. In general, for each additional feature you want to track, the existing number of compartments needs to be replicated by the discrete categories you have. For gender, one would need to have 2x the compartments. Similarly if one wanted to stratify according to young/medium/old age, 3x the compartments are required, etc.

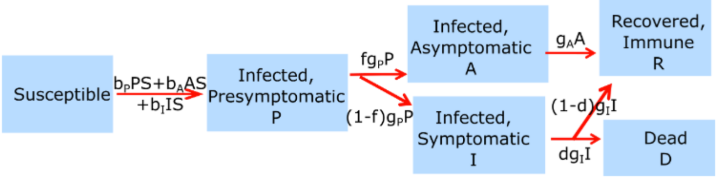
In addition to specifying the *compartments* of a model, we need to specify the dynamics determining the changes for each compartment. In general, more compartments leads to more processes and more parameters governing these processes.

For this model, we include the following processes:

- Susceptible individuals (S) can become infected by pre-symptomatic (P), asymptomatic (A) or symptomatic (I) hosts. The rates at which infections from the different types of infected individuals (P, A and I) occur are governed by 3 parameters, b_P, b_A, and b_I. In other words, those b_i parameters determine how infectious an individual in stages P, A and I is.
- All infected individuals first enter the presymptomatic stage. They remain there for some time (determined by rate g_P, the inverse of which is the average time spent in the presymptomatic stage). A fraction *f* of presymptomatic hosts move into the asymptomatic category, and the rest become symptomatic infected hosts.
- Asymptomatic infected hosts recover after some time (specified by the rate g_A).
- Similarly, infected symptomatic hosts leave that stage at rate g_I. For symptomatic hosts, two outcomes are possible, either recovery or death. The parameter *d* determines the fraction of hosts that die.
- Recovered individuals are immune to reinfection.

Model Implementation

The flow diagram and the set of ordinary differential equations (ODE) which are used to implement this model are as follows:



Flow diagram for this model.

$$\dot S = -S(b_P P + b_A A + b_I I)$$

$$\dot P = S(b_P P + b_A A + b_I I) - g_P P$$

$$\dot A = f g_P P - g_A A$$

$$\dot I = (1-f) g_P P - g_I I$$

$$\dot R = g_A A + (1-d) g_I I$$

$$\dot D = d g_I I$$

DSAIDE tasks

Task 1

- Set the simulation with 1000 susceptibles and 1 infected.
- Simulation time 12 months, *g*=5, *b*=0.01.
- Run the simulation, you should get an outbreak. Use the final size equation linking R_0 and the fraction of susceptible hosts left at the end of the outbreak to compute the reproductive number.

Task 2

- Use the equation that expresses R_0 as a function of model parameters for the simple SIR model. Using the values of the model parameters, compute R_0 that way. Check that it agrees with what you found in the previous task.

Task 3

- Double the value of the transmission parameter, *b*. Leave everything else as before.
- Before you run the simulation, what do you expect to see and what do you expect to get for R_0?
- Run the simulation and compute R_0 using the final outbreak size to test your expectations.

Task 4

- Double the rate of the recovery parameter, *g*. Leave everything else unchanged.
- Think about your expectations for R_0 and the resulting outbreak dynamics.
- Run the simulation to check your expectations. Use the final outbreak size to compute R_0.

Task 5

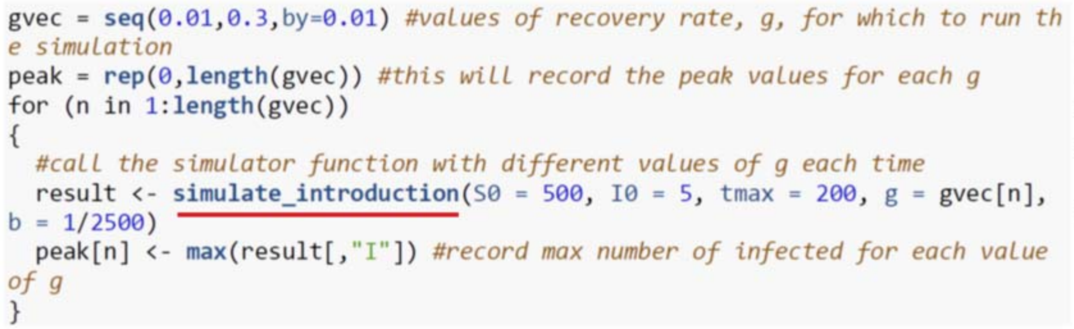
- Set the model parameters back to those given in task #1.
- Another way to estimate R_0 is to determine the rate of increase in infected hosts at the beginning of the outbreak. During the initial phase, new infections increase exponentially according to $I(t)=I_0 \exp(rt)$, with *r* being the rate of growth. Usually, for any real outbreak, you do not know the number of infected at the start, I_0, or the exact time the outbreak starts. It is still possible to estimate *r* by obtaining two values of *I* at two time points during that initial growth rate, i.e. I_1 at time t_1 and I_2 at time t_2. One obtains equation $I_1=I_0 \exp(r\ t_1)$ and $I_2=I_0 \exp(r\ t_2)$. By solving one of these equations for I_0 and substituting into the other, we get $I_2= I_1 \exp(r\ (t_2 - t_1))$. By solving the model for *r* and entering numbers for I_1 and I_2 and times t_1 and t_2 we can figure out *r*.
- Let's try that. Run the model with tmax = 0.2 and tmax = 0.4 and record the number of infected at the end of the simulation for each time. Then substitute all the values into the equation you found for *r* and thus compute the growth rate.
- For this model, the growth rate and R_0 are related through $R_0 = 1+rD$, where *D* is the average duration of the infectious period (i.e. the inverse of the recovery rate). Use this to determine R_0. You should get essentially the same answer (up to some rounding differences) as for task #1.
- Note that the choice of t_1 and t_2 can influence the results. Earlier times are better since once the number of susceptibles starts to drop markedly, the growth of infected slows down and is not exponential anymore.

Task 6

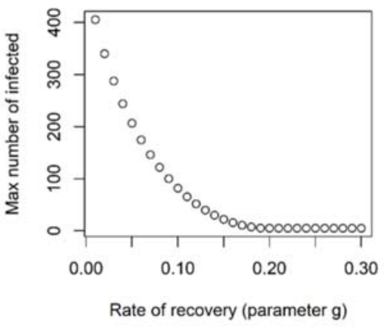
- What is the value of the reproductive number *R* at the time the outbreak peaks? (It's only called R_0 at the beginning for a fully susceptible population). Explain how you can find that value for R, both using intuitive reasoning and using the equation for R_0 given above (R_0 = 1+rD). Note that at this R value, the outbreak wanes, but people still get infected. What R value would you need to halt any further infections completely?

Easy advancement

- Students can advance from the graphical exploration of the models (Level 1) to adding a bit of their own code and make the models do more (L2) all the way to using the model code and modifying it to fit their needs (L3).



```
gvec = seq(0.01,0.3,by=0.01) #values of recovery rate, g, for which to run th
e simulation
peak = rep(0,length(gvec)) #this will record the peak values for each g
for (n in 1:length(gvec))
{
  #call the simulator function with different values of g each time
  result <- simulate_introduction(S0 = 500, I0 = 5, tmax = 200, g = gvec[n],
b = 1/2500)
  peak[n] <- max(result[,"I"]) #record max number of infected for each value
of g
}
```



Other considerations

- The software is written as R package. R is a powerful and FREE, widely used statistical and programming language.
- The packages are open source and publicly developed on Github and CRAN.
- The packages are developed such that students can seamlessly move from graphical interaction (exploring models) to doing their own coding (becoming modelers).

More tools

- DSAIDE/DSAIRM work well for exploring models that I pre-wrote.
- DSAIDE/DSAIRM are written such that users can go beyond the graphical interface and gain flexibility without too much additional coding.
- **However**, if a user wants to build/explore new models, they usually have to take the (pre-written) models and alter them. Better than starting from scratch, but still requires coding.
- A new R package, called *modelbuilder* allows individuals to graphically build and analyze custom compartmental (ODE/stochastic/discrete-time) without the need to write code.
- Package is in development, current version available at: https://ahgroup.github.io/modelbuilder/

DSAIDE in action

- If you haven't done yet, follow the brief installation instructions here: https://ahgroup.github.io/DSAIDE/
- I strongly recommend installing DSAIDE as R package. If for some reason that does not work, you can access it online here: https://epibiouga.shinyapps.io/dsaide/
- Start with the *Basic SIR Model* app.
- Continue with the *ID control for multiple outbreaks* app.
- Continue to explore any app you are interested in. For advanced users, you can download the simulation functions for all apps and try the L2 and L3 approach. See the "Get Started" tutorial on the package website for examples.